ニューラル記号推論における推論過程の教示方法

青木洋一 [1] 工藤慧音 [1] Ana Brassard[2,1] 栗林樹生 [1,3] 吉川将司 [1]
坂口慶祐 [1,2] 乾健太郎 [1,2]
[1] 東北大学 [2] 理化学研究所 [3]Langsmith 株式会社
{youichi.aoki.p2,keito.kudo.q4}@dc.tohoku.ac.jp,
ana.brassard@riken.jp,kuribayashi@tohoku.ac.jp
{yoshikawa,keisuke.sakaguchi,kentaro.inui}@tohoku.ac.jp

概要

ニューラルモデルを用いた記号推論能力は推論過程を生成することで向上する．しかし，推論過程の性質や形式がどのような影響を与えるかは分かっていない．本研究では，推論過程について出力戦略と推論戦略という 2 つの軸から，ニューラルモデルで形式的な推論を行う際の適切な推論過程の教示方法を探る．我々は多段数値推論問題（A=1，B=3，C=A+3，C?）を用いた統制的実験を行い，各形式の選択が性能に大きな影響を与える事が分かった．加えて，少なくとも本実験設定範囲内では，ニューラルモデルによってほぼ完璧な推論が達成され，適切な戦略の選択が重要である事を示す．

1 はじめに

ニューラルネットワークという「柔らかい」道具立ての上で記号的推論が実現可能かについては，人工知能分野における根幹的な問いである．本研究では，ニューラル系列変換モデルで多段数量推論を実現できるのかという例題のもと，適切な推論過程の教示方法を選択することで，少なくとも我々の設定内では記号推論が実現可能であることを示す．

近年，ニューラルモデルに推論過程を生成させることで，数量推論 [1, 2, 3, 4, 5]，常識推論 [1, 6]，記号推論問題 [1, 3] など様々な設定において性能が向上することが示された．しかしながら，どの様に推論を行うかなどは暗黙的に決定されるなど，しばしば限定的な設定が課されている．

実験では推論過程の教示方法について，**出力戦略**と**推論戦略**の 2 軸に分解し，ニューラルモデルに形式的な推論を行わせる際の適切な設定を調査する（図 1）．出力戦略（§2.2）は，推論過程生成の粒度（一遍出力, ステップ出力, 文字出力）を指す．既存研究では様々な方法が採用されており，統制的な分析はなされていない．例えば，推論過程とそこから導かれる結論を一度に生成させる形式 [1, 3, 4, 5, 6, 7]，推論過程をある程度生成し，結論が出るまで繰り返す形式 [8, 9, 10]，推論過程と共にサブゴールも繰り返し生成する形式などがある [11, 12]． 推論戦略（§2.3）は推論を行う順番・内容（最短経路, 全探索経路, 後向き＋最短経路）を指す．例えば，質問内容から逆向きに手がかりを探索し，見つけた手がかりを用いて推論を行う手法や（後向き＋最短経路）[9, 13, 14]，与えられた文脈に対して網羅的に推論を行う形式もある [10, 11, 15]．

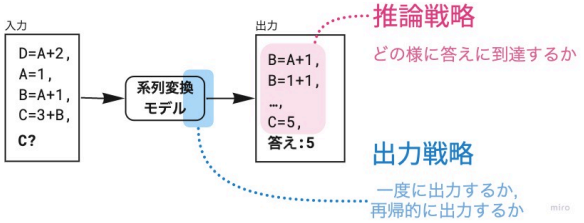
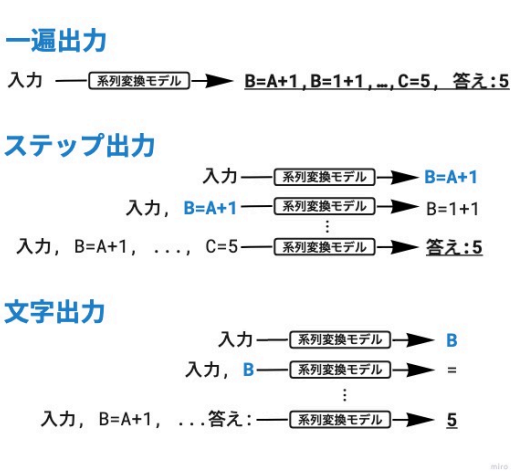


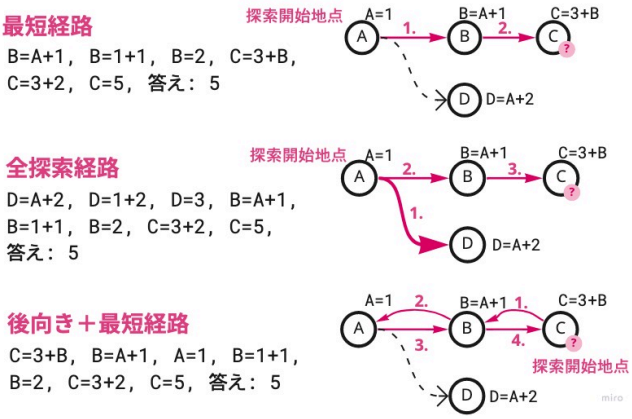
図 1: 出力戦略と推論戦略の選択が，ニューラルモデルの多段推論の学習可能性に大きく影響する．

数量推論問題を通して，ニューラルモデルで記号推論を学習する際にどの様な戦略の組み合わせが有効を調査した．ただし，この形式言語によって構成された純粋な設定は自然言語上においても記号推論を行うための必要条件を明らかにする事を目的としている．実際，この純粋な数量推論問題が解けないモデルが，より複雑な問題で適切な汎化を達成するとは期待できない．

実験の結果，各形式の選び方がニューラル系列変換モデルの記号推論性能に大きく影響することを発見し，少なくとも本実験設定範囲内では，推論の深さ方向の外挿に対してほぼ完璧な汎化を達成する戦略の組み合わせがある事を明らかにした．



(a) **一遍出力**: 一度の入力に対して，推論過程全体と答えを出力する．**ステップ出力**: 1 つの推論ステップを繰り返し出力する．**文字出力**: 1 **文字**を繰り返し出力する．



(b) グラフのノードが変数，エッジが依存関係を表す．**最短経路**: 答えの導出に必要不可欠な式のみを出力する最小の推論過程．**全探索経路**: 目標に到達するまで，貪欲に式を解く．**後向き＋最短経路**: 質問内容から逆向きに手がかりを探索し（図中ステップ 1,2），見つけた手がかりを用いて最短経路を出力（図中ステップ 3,4）．

図 2: 入力:D=A+2，A=1，B=A+1，C=3+B，C?が与えられた場合の（a）出力戦略（b）推論戦略

2 実験設定

2.1 問題定義

D=A+2，A=1，B=A+1，C=3+B，C?という方程式が与えられ，任意の変数の値（例えば C の値）を答える問題を用いた（図 1）．各式は代入（A=1）またはモジュール加算と代入（B=3+1）で構成される．ただし，モジュール加算は mod 100 である．問題の文脈には，答えを計算するのに必要でない式も含まれている．（図 1 中の D=A+2）．特定の変数に割り当てられた値は通常，異なる式で参照される（A=1，B=A+1）．また，式の順番，数字や変数の値は無作為に割り当てられている．各問題には推論の深さ（答えに到達するために**必要な**式の数）が定義されている．例えば，A=1，B=2+A，C=3+B，D=2，C? という問題の推論の深さは 3（A=1，B=2+A，C=3+B）である．

人工データを用いる動機　DROP [16] などといった自然言語で記述されたデータではなく，本人工データを用いた理由は 3 点ある．第一に，汎化能力の検証のため推論の深さといった問題の複雑さの統制が容易である点が挙げられる．一般的な数量推論問題の場合，推論の深さの統制が難しい（例えば，推論の深さが 10 である様々な事例を考え出すのは容易でない）．実験では推論の深さ方向の外挿に焦点を当て，深さが浅い（深さ 1–5）事例を使ってモデルを学習し，深さが浅い／深い（深さ 1–12）事例で評価する．

第二に，自然言語による記述は暗黙のうちに「偽のバイアス」を与える可能性があり，モデルが高い性能を示した際に，意図した解き方で解いているのか解釈が難しい [17, 18, 19, 20, 21, 22]．我々のデータでは，記号の出現の偏りなどを排除していることと，数量推論であり解空間が広いことから，モデル誤った汎化により正答する可能性や偶然正答する可能性を排除できている．

第三に，我々の設定は数量推論問題を解くための必要条件であるという点が挙げられる．

2.2 出力戦略

既存研究に従い系列変換モデルを用いる [8, 10, 11]．次の 3 つの出力（デコード）形式を比較した: 一遍出力，ステップ出力，文字出力（図 2a）．

一遍出力: 入力に対して，推論過程全体と答えを一度に出力する．（chain-of-thought と同様の形式）[1, 2, 12, 23]．この設定では，推論過程数が長いほどデコーダが一度に生成しなければならない系列長が長くなる．

ステップ出力: 入力に対して，推論 1 ステップを出力する．元の入力に生成された出力ステップを書き足して再度エンコーダに入力し，次の推論ステップを出力させる．この処理を停止文字が出力

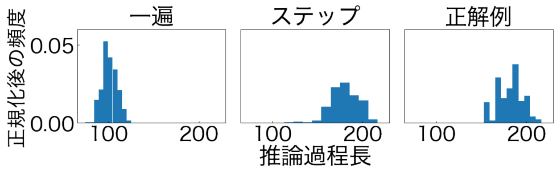


図 3: 推論の深さ 12 の問題に対して，一遍出力，ステップ出力が生成した推論過程長（文字数）の分布．

されるまで，あるいは設定した最大反復回数（100）に達するまで繰り返す（proofwriter と同様の形式）[8, 9, 10, 11, 12].

文字出力: 入力に対して，1 つの文字だけを出力する．ステップ出力と同様に，元の入力に出力文字を追加し，推論が停止するまで処理を繰り返す．最大反復回数は 500 に設定した．

一遍出力では，問題の複雑さ（推論の深さ）に応じて生成すべき系列の長さが変化し，ステップ，文字出力では問題の複雑さに関係なく，生成長（1 推論ステップ/文字）はほぼ変わらない．また，ステップ出力と文字出力を比較すると，問題を意味のある単位で分割することの優位性が検証できる．

2.3 推論戦略

任意の変数を導出するための推論経路は複数考えられる．したがって，既存研究に関して，我々は 3 つの推論戦略を比較した．**最短経路**，**全探索経路**，**後向き＋最短経路**（図 2b）.

最短経路: 答えの導出に必要不可欠な式と，それら用いた推論結果のみを出力する形式（図 2b の例では D の値を使わずに C の値を導出可能なため，D に関する推論が推論過程に含まれていない）[1, 2, 12, 23].

全探索経路: 問いに関連する式であるかは考えず，答えに達するまで与えられた方程式を順に貪欲に解き続ける形式 [10, 11, 15]．具体的には，各ステップで最も左にある解ける方程式を計算する．ただし，この方法は長い推論過程を導くことになるため計算量の観点からは非効率である．

後ろ向き＋最短経路: 質問内容から逆向きに手がかりを探索し（図中ステップ 1,2），見つけた手がかりを用いて最短経路を出力する形式 [9, 13, 14].

推論過程無し: 基準の設定として，推論過程を生成させず直接答えを出力させる形式も検証した．

3 実験

3.1 実験設定

モデル: 我々は事前学習済みの T5-base を使用した [1] [24].

学習: まず 10K の基本的な演算（すなわち，代入，参照，加算）に関する事前学習用データセットを用いて 30 エポックの事前学習を行う．具体的には，A=1，A?）や，A=1+3，A?）のような形式のデータセットを用意した．次に，深さ 1–5 で合計 5K の学習データ（各推論の深さに対して 1K 個の学習事例）を用いて 2000 エポックの学習を行った．詳細な実験設定は付録 A に記述する．

評価: 未知の問題への汎化能力を評価するため，比較的単純な問題（深さ 1–5）での評価に加え，より複雑な問題（深さ 6–12）における性能を評価する．評価は各推論の深さに対して 200 個の評価事例で行った．

3.2 結果：出力戦略

推論戦略を最短経路に固定し，出力戦略を比較した．図 4a に推論の深さごとの正解率を示す．なお，ここでの正解率は答え（例えば，C=6）が正しいかどうかで決定した．その結果，(i) **推論過程を生成すると性能が向上する**，(ii) 出力戦略の中では，**ステップ出力が最も良い性能を導く**かつ**一遍出力の性能が最も悪い**ことが分かった．

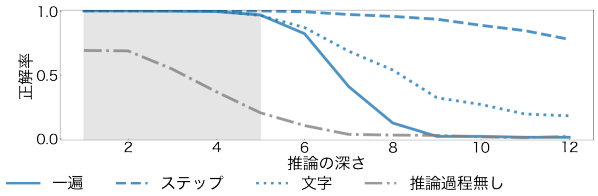
一遍出力の性能が劣る原因は，デコーダが（浅い）学習データと同じような長さの推論過程を出力するように過学習したためであると予想を立てた．実際，一遍出力を用いた場合，領域外（推論の深さ 12 など）の設定でも比較的短い推論過程を生成しており（図 3），この仮説は支持される．

文字出力に対するステップ出力の優位性は，推論過程を意味のある単位に分割し，各ステップをエンコーダーデコーダーの 1 回の呼び出しでモデル化することの優位性を示唆している．

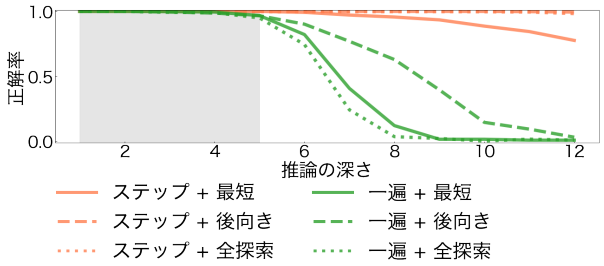
3.3 結果：推論戦略

図 4b は，出力戦略をステップ出力に固定した場合の各推論戦略における，推論の深さ毎の正解率である．推論の深さが増すと最短経路の設定では性能

1) https://huggingface.co/docs/transformers/model_doc/t5



(a) 出力戦略



(b) 推論戦略

図 4: 推論の深さに対するモデルの正解率．灰色の範囲は学習領域（深さ 1–5）を示している．図 4a は，一遍出力を用いた場合の推論過程長の増加に伴う正解率低下を表している．図 4b は，ステップ出力と後向き＋最短経路または全探索経路の組み合わせにより汎化がうまくいく事を示している．

表 1: ステップ出力，最短経路設定における誤りの例．（スキップ）は，推論ステップが誤ってスキップされたことを表す．

Question: A=1，B=2+A，B?

誤りの種類	正解例	モデルの出力
コピー誤り	**B=2+A，** B=2+1， B=3	**B=6+A，** B=6+1， B=7
早すぎる代入	**B=2+A，** B=2+1， B=3	（スキップ） B=2+2， B=4

が低下するが，全探索経路または後向き＋最短経路では，推論の深さ 6-12 に外挿した場合であっても正しい推論を行えた．また，全探索経路と後向き＋最短経路では，最後の答えだけでなく，間の推論過程も正しく生成された（約 100% の正解率）．詳細は付録（表 2）に記述するなお，これらの形式は出力戦略を一遍出力にした場合では効果がなかった．

先行研究では推論過程長が長い場合，推論過程を**生成しない**モデルの方が生成するモデルより汎化性能が高いというやや非直観的な結果が報告されているが [25]，我々の結果は適切な出力戦略を選択することで推論過程の生成は効果的であることを示唆している．

最短経路が劣る原因は，推論過程の情報が不十分なことにあると考えた．モデルは推論過程を出力する前に最短経路を知ることができない．そのため，最短経路であっても最短経路上の変数以外も探索する必要がある．この処理を最短経路は暗黙のうちに行っている．一方，図 2b から分かる様に，全探索経路はこの過程を明示的に行なっている．そのため，たとえ推論過程が長くなったとしても，**後向き＋最短経路や全探索経路など，推論過程をくまなく提示した方が正解率が高くなる**と結論づけた．

3.4 誤り分析

最短経路における推論の深さ 12 の事例の誤りを分析した．[2)]（i）コピー誤り，（ii）早すぎる代入という 2 種類の誤りが確認され，表 1 にそれぞれの誤りの例とその割合を示す．最も頻度の高いもの（53%）は単純なコピー誤りで，モデルが元の方程式を推論過程に正確にコピーできなかったというものである．誤ったコピー能力は先行研究でも示されており [26]，モデルにコピー機構を導入する利点が支持される [27]．次に，早すぎる代入は，モデルが文脈から方程式をコピーするステップを飛ばして，無作為に値を代入する誤りである．これらの誤りは最短経路以外の推論戦略ではほぼ解決され，学習時に最短経路で提示している情報が不十分であることが示唆される．

4 おわりに

記号推論におけるニューラル系列変換モデルの推論過程の教示方法を調査し，ステップ出力と細かな粒度の推論を組み合わせることで記号推論をより良く行う事ができることを明らかにした．この結果は，ニューラルモデルによるより確実な記号推論の実現可能性を支持するものである．

しかしながら，この傾向がより複雑な記号的推論や自然言語で書かれた問題に対して一般化されるかどうかは不明である．したがって，異なる設定において一般化しない場合，我々はそのギャップを縮めて行くことを目指す．

2) 合計 32 の事例を解析した．これは 1 つのシードにおける全不正解数である．

謝辞

本研究は JST CREST JPMJCR20D2 及び JSPS 科研費 JP22H00524,21K21343 の助成を受けたものです.

参考文献

[1] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. **CoRR**, 2022.

[2] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. **CoRR**, 2021.

[3] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. **CoRR**, 2022.

[4] Gabriel Recchia. Teaching autoregressive language models complex tasks by demonstration. **CoRR**, 2021.

[5] Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. **CoRR**, 2022.

[6] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. **CoRR**, 2022.

[7] Maxwell I. Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models. **CoRR**, 2021.

[8] Soumya Sanyal, Harman Singh, and Xiang Ren. Fairr: Faithful and robust deductive reasoning over natural language. In **ACL 2022**.

[9] Gabriele Picco, Thanh Lam Hoang, Marco Luca Sbodio, and Vanessa López. Neural unification for logic reasoning over natural language. In **EMNLP 2021**.

[10] Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. Proofwriter: Generating implications, proofs, and abductive statements over natural language. In **ACL/IJCNLP 2021**.

[11] Zhengzhong Liang, Steven Bethard, and Mihai Surdeanu. Explainable multi-hop verbal reasoning through internal monologue. In **NAACL-HLT 2021**.

[12] Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Unsupervised commonsense question answering with self-talk. In **EMNLP 2020**.

[13] Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving. In **NeurIPS 2017**.

[14] Nuri Cingillioglu and Alessandra Russo. Deeplogic: Towards end-to-end differentiable logical reasoning. In **AAAI 2019**, CEUR Workshop Proceedings.

[15] Kaiyu Yang, Jia Deng, and Danqi Chen. Generating natural language proofs with verifier-guided search. **CoRR**, 2022.

[16] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In **NAACL-HLT 2019**.

[17] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In **NAACL-HLT,2018**.

[18] Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. BERT & family eat word salad: Experiments with text understanding. In **AAAI 2021**.

[19] Hadeel Al-Negheimish, Pranava Madhyastha, and Alessandra Russo. Numerical reasoning in machine reading comprehension tasks: are we there yet? In **EMNLP 2021**.

[20] Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. What makes reading comprehension questions easier? In **EMNLP 2018**.

[21] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. In **EMNLP 2017**.

[22] Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In **ACL 2019**.

[23] Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, Nitish Shirish Keskar, and Caiming Xiong. Modeling multi-hop question answering as single sequence prediction. In **ACL 2022**.

[24] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. **J. Mach. Learn. Res.**, Vol. 21, pp. 140:1–140:67, 2020.

[25] Nicolas Gontier, Koustuv Sinha, Siva Reddy, and Christopher Pal. Measuring systematic generalization in neural proof generation with transformers. In **NeurIPS 2020**.

[26] Song Xu, Haoran Li, Peng Yuan, Youzheng Wu, Xiaodong He, and Bowen Zhou. Self-attention guided copy mechanism for abstractive summarization. In **ACL 2020**.

[27] Santiago Ontanon, Joshua Ainslie, Zachary Fisher, and Vaclav Cvicek. Making transformers solve compositional tasks. In **ACL 2022**, Dublin, Ireland.

[28] Jeonghwan Kim, Giwon Hong, Kyung-min Kim, Junmo Kang, and Sung-Hyon Myaeng. Have you seen that number? investigating extrapolation in question answering models. In **EMNLP 2021**.

[29] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In **ACL 2020**.

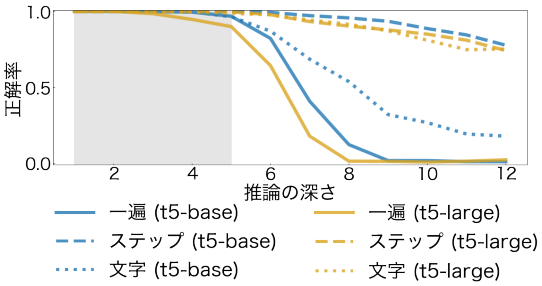


図 5: 推論の深さに対する T5-base と T5-large の正解率の変化.

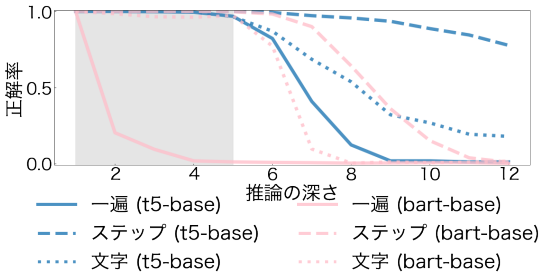


図 6: 推論の深さに対する T5-base と BART-base の正解率の変化.

A　詳細な実験設定

学習率は，10^{-3}，10^{-4}，10^{-5} のうち，学習データでの損失の収束が最も早い学習率を使用した．モデルに関しては学習後のエポック毎のチェックポイントのうち，損失が最も小さいモデルを使用した．また，節 3 の実験結果は 3 種類のシード値で得られた結果の平均値である．トークナイザは，`0` から `9` までの全ての数値を保持し，我々のデータセットに含まれる数値は予め桁ごとに分割される（“12” は “`@@1 @@2`” の様に分割される） [28].

B　モデルサイズが異なる場合

モデルサイズが異なる場合の設定として，t5-large と t5-base を比較した．図 5 にその結果を示す．T5-large は T5-base と比較して，一遍出力とステップ出力の正解率がほぼ同等であることがわかる．一方，文字出力の場合，t5-large の方が正解率が高い．この結果から，文字単位の出力は，モデルサイズを大きくする必要がある事がわかった．

C　アーキテクチャが異なる場合

アーキテクチャが異なる場合の設定として，BART-base [3] [29] を基準として，T5 の NLP タスクによる事前学習の有効性を調査した．図 6 はこの結果を示し，T5 が BART より優れていることがわかる．これは，NLP タスクの事前学習が記号推論にも有効であることを示唆している．

D　推論過程の正解率と誤り分析

表 2 は推論過程の正解率を示している．ここで推論過程の正解率（スコアの左側）は完全一致ではなく，数学的な正しさを基準に評価したものである．

表 2: 各推論の深さにおけるステップ出力を用いた T5-base の正解率（推論過程／答え）．推論過程の正解率（スコアの左側）は，完全一致を基準に測定した．

推論の深さ	最短	後向き＋最短	全探索
6	100.0/100.0	100/100	100/100
7	99.5/100.0	100/100	100/100
8	99.0/ 99.0	100/100	100/100
12	92.5/ 92.5	100/100	100/100

また，答えが正しく，推論過程が誤っている場合を分析した結果，2 つの誤りパターンが見受けられた．1 つ目は，推論過程の誤り部分が答えの導出には影響が無い場合が挙げられる． 具体的には，`D=3+A`，`A=1`，`D=3+2`，`D=5` の様に D の値を誤って導出したとしても答えの導出に D の値を用いない場合，正しい答えを出力する事があった．2 つ目は，一度誤った推論過程が再び誤った推論をするなどして，最終的に正しい推論過程に合流する場合が挙げられる．具体的には，変数 `B` の値を求める際に `B=2+D` という誤った推論ステップが出力された後，正しい推論ステップ `B=2+A` が出力されるような事例を確認する事ができた．

3) https://huggingface. co/docs/transformers/model_doc/bart